Docket No.: M1103.70273US00
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Michael A. Tavis
Serial No.:	10/032,709
Confirmation No.:	5403
Filed:	October 24, 2001
For:	METHOD AND APPARATUS FOR MANAGING SOFTWARE COMPONENT DOWNLOADS AND UPDATES
Examiner:	Bilgrami, Asghaar H.
Art Unit:	2443

Certificate of Electronic Filing Under 37 CFR 1.8

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being transmitted via the Office electronic filing system in accordance with § 1.6(a)(4).

Dated: July 16, 2009 Signature: Danielle Calder

APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Madam:

As required under § 41.37(a), this brief is filed in furtherance of the Notice of Appeal filed in this case on April 16, 2009. Extensions of time are addressed in the transmittal accompanying this brief.

This brief contains items under the following headings as required by 37 C.F.R. § 41.37 and M.P.E.P. § 1205.2:

I.	Real Party In Interest
II	Related Appeals and Interferences
III.	Status of Claims
IV.	Status of Amendments
V.	Summary of Claimed Subject Matter
VI.	Grounds of Rejection to be Reviewed on Appeal
VII.	Argument
VIII.	Conclusion
	Claims Appendix
	Evidence Appendix
	Related Proceedings Appendix

I. REAL PARTY IN INTEREST

The real party in interest for this appeal is:

Microsoft Corporation

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

A. Total Number of Claims in Application

There are 40 claims pending in the application.

B. Current Status of Claims

1. Claims canceled: 31
2. Claims withdrawn from consideration but not canceled: none
3. Claims pending: 1-30 and 32-41
4. Claims allowed: none
5. Claims rejected: 1-30 and 32-41

C. Claims On Appeal

The claims on appeal are claims 1-30 and 32-41.

IV. STATUS OF AMENDMENTS

Applicant filed an Amendment After Final Rejection on January 16, 2009. The Examiner responded to the Amendment After Final Rejection in an Advisory Action mailed February 3, 2009. In the Advisory Action, the Examiner indicated that Applicants' proposed amendments to claims 1-30 and 32-41 would be entered.

Accordingly, the claims enclosed herein as Appendix A incorporate the amendment to claim 21, as indicated in the Amendment After Final Rejection filed January 16, 2009.

V. SUMMARY OF CLAIMED SUBJECT MATTER

As an overview, the present application describes a peer-to-peer collaboration system. Such a system operates by creating local copies of a data model, called a shared "space," in computers used by each member in a collaboration session. "Activity" software in each user computer interacts with the user to receive input about changes to the shared space. The activity then distributes these changes to the computers of the other users who are participating in the collaboration session. Upon receipt at other users' computers, the activities in those computers process the changes to make changes to the shared spaces maintained by those computers [0002].

The effect of this interaction is that each user's computer has a consistent copy of the shared data space, which may be used in collaboration between users. The application describes that an activity may maintain data used in games, such as chess or tic-tac-toe [0003]. Though, it should be recognized that games are only an example and a peer-to-peer collaboration system could be used in other instances where multiple users want to concurrently view and change the same data.

The "activity" software can contain multiple components that allow the activity to perform the desired functions [0004]. The present application solves a problem that can arise in a peer-to-peer collaboration system. In order to maintain consistent copies of the shared space for each of the users, each user computer is configured with components that lead to execution of each change request in the same way. However, maintaining the user computers to have consistent sets of components may be difficult because users may not even be aware that other users have upgraded components [0005-0006]. Further, the peer-to-peer model makes it difficult to automate the upgrade process. In the peer-to-peer model, there is no central location with which all users must interact so there is no central location from which component upgrades can be released [0008].

The application describes a peer-to-peer collaboration system in which components may be updated in response to an indication that a component is in use in a collaboration session [0049] or in response to some action of a user participating in the collaboration session [0047]. In this way, components can be upgraded even if the user is unaware of the need for the upgrade. Further, these upgrades can be made unobtrusively, such as while the user is waiting for the collaboration system to perform other actions [0006].

Turning specifically to the claims:

A) Independent Claim 1

Claim 1 is directed to an apparatus adapted for use in a peer-to-peer collaboration system having a plurality of peer devices. The apparatus comprises a computer system with a memory and a computer-readable medium having computer executable modules [0004, 0124]. Included among the computer-executable modules is an activity program adapted to implement a portion of a collaboration session on a first peer device. The activity program can modify a local data copy of a shared space in response to deltas generated as a result of user actions within the collaboration system at the first peer device and other peer devices of the plurality of peer devices [0002]. The activity program also can generate a component update request in response to an action by a user within the collaboration session that indicates a change to the shared space made with the component [0047, 0002]. Another module is a component manager that receives the component update request from the activity program and has a parser that extracts from the request URL information which identifies the location of a file containing software component resources for satisfying the component update request [0056, 0063]. A download manager receives the URL information from the component manager and has a file retriever which asynchronously retrieves the file from the specified location and places the file in a staging area in the memory [0073, 0076]. An install manager asynchronously installs the file [0106]. The component manager is also adapted to determine whether the requested software component is already installed on the computer system and to selectively invoke the download manager based on the determination [0063].

B) Independent Claim 11

Claim 11 is directed to a method of operating a computer system with a memory [0124] as part of a peer-to-peer collaboration system comprising at least one other computer system maintaining a first local data copy of a shared space [0002]. The method comprises: (a) generating a component update request in response to receiving information about a component being used in a collaboration session involving the at least one other computer [0032], the component update

request identifying the component, and the received information being generated dynamically while the collaboration system is in operation to modify the shared space based on input of a plurality of users collaborating using the peer-to-peer collaboration system [0047, 0002]; (b) parsing the request to extract from the request URL information which identifies the location of a file containing software component resources for satisfying the request and an identification of the component [0056, 0063]; (c) determining, based on the identification of the component, availability of the component; (d) selectively in response to the determination, using the URL information to asynchronously retrieve the file from the specified location [0073, 0076]; and (e) asynchronously installing the component from the file [0063], whereby the computer system can use the component to maintain a second local data copy of the shared space that is synchronized with the first local data copy [0032].

C) Independent Claim 21

Claim 21 is directed to a computer program product stored on a computer readable medium for use in a peer-to-peer collaboration system including a computer system with a memory [0004, 0124]. The computer system is adapted to receive an update request generated as a result of user interaction with the peer-to-peer collaboration system by a plurality of members of a collaboration session [0047, 0002]. The computer program product comprises program code for running a process that maintains a local copy of a shared space, the process being adapted to respond to input from the plurality of members and generate a request to update a component used in modifying the local copy of the shared space, the request being generated in response to user input indicating use of the component [0047, 0002]. Program code extracts from the request information which identifies the location of a first file for satisfying the request [0056, 0063]. Program code uses the information to retrieve the first file [0073, 0076]. Program code extracts from the first file an indicia of a trusted supplier and obtains second location information of a second file, the second file containing a second component [0055]. Program code uses the second location information to retrieve the second file [0092]. Program code extracts from the second file the second component and an indicia of a supplier of the second component [0091-93]. Program code selectively installs

the second component when the indicia of the supplier is consistent with the indicia of a trusted supplier [0077, 0088-91, 0093].

D) Independent Claim 32

Claim 32 is directed to an apparatus for use in a peer-to-peer collaboration system. The system has a computer system with a memory [0124] and at least one other computer system maintaining a first local data copy of a shared space [0002]. The apparatus comprises means for implementing a collaboration session for a user. The means for implementing is adapted to maintain a local copy of a data space shared by a plurality of collaborating members in the collaboration session and to receive an indication of a component in use within the collaboration session involving the at least one other computer and to selectively generate an update request for the component based on the indication of use within the collaboration session [0002, 0047]. Means responsive to the request parses the request to extract from the request URL information which identifies the location of a file containing software component resources for satisfying the request[0056, 0063]. Means receives the URL information and asynchronously retrieves the file from the identified location [0073, 0076]. Means, cooperating with the parsing means, installs the component from the file while the collaboration system is executing [0106], whereby the computer system can use the component to maintain a second local data copy of the shared space that is synchronized with the first local data copy [0032, 0036].

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether independent claim 1 is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether claims 2, which depends from claim 1, is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether claim 8, which depends from claim 2, is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether claim 9, which depends from claim 1, is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether independent claim 11 is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether claim 12, which depends from claim 11, is properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Whether independent claim 21 is properly rejected under 35 U.S.C. §103(a) based on Varma et al. (U.S. 6,334,141) and Carley et al. (U.S. 6,701,345).

Whether independent claim 32 is properly rejected under 35 U.S.C. §103(a) based on Varma et al. (U.S. 6,334,141) and Carley et al. (U.S. 6,701,345).

Arguments for reversal for each of the above identified grounds for review are presented separately below. Separate arguments are not provided for dependent claims 3 through 7 and 10, which stand or fall with claim 1. Separate arguments are not provided for dependent claims 13 through 20, which stand or fall with claim 11. Separate arguments are not provided for dependent claims 22 through 30, which stand or fall with claim 21. Separate arguments are not provided for dependent claims 33 through 41, which stand or fall with claim 32.

VII. ARGUMENT

Applicant respectfully requests that the Examiner's final rejection of all the claims be reversed.

A. Prior Art Cited

All of the claims on appeal are rejected based on one of two different combinations of references. These combinations are Donohue et al. (U.S. 6,199,204) combined with Parthesarathy et al. (U.S. 6,353,926) or Varma et al. (U.S. 6,334,141) combined with Carley et al. (U.S. 6,701,345). However, these references, whether considered alone or in combination, do not describe a peer-to-peer collaboration system that performs component upgrades based on user actions related to a collaboration system, as in the present application.

1. Overview of Donohue et al.

Donohue describes an updater for use in updating a computer program installed on a computer system. The updater includes information that identifies one or more locations on a network where software resources for updating the program are located. The updater automatically controls obtaining these resources and applying them to fix bugs within the software program (col. 3, lines 50-67). These updates are performed according to predefined criteria (see, column 4, lines 19-21; col. 15, lines 60-62).

Donohue describes that an updater component can interact with other updater components on the same computer so that, if an update to a first program has as a prerequisite that an update to a second program be performed, the updater for the first program can cause the updater for the second program to perform an update (col. 6, lines 24-32). If an updater to perform the prerequisite update is on a different computer than the first program, an updater for the first program can contact an updater for the second program on another computer (col. 13, line 55- col. 14, line 2).

2. Overview of Parthesarathy et al.

Parthesarathay describes a method for a software vendor to notify a user that a software update is available (Abstract). According to this method, a software vendor establishes a software update channel that includes a link between the software vendor's website and the user computer. Once a user subscribes to the update channel, the user's computer will periodically check for available updates (col. 5, line 61 - col. 6, line 6). If an update is available, the user may be asked whether the update should be installed or it could be automatically installed (col. 6, lines 16, 30).

3. Overview of Varma et al.

Varma describes a distributed server for real-time collaboration (Title). The distributed server substitutes for a centralized server (col. 1, lines 9-10). The role of the centralized server is to collect and order changes to a shared work space. These changes may then be distributed from the centralized server to the clients (col. 1, lines 39-48).

Varma describes an approach of breaking the workspace into disjoint partitions that can be independently modified (col. 3, lines 52-55). Changes related to each partition can be handled by independent servers (col. 3, lines 58-60). As a result, a bottleneck associated with a single, centralized server is avoided (col. 2, lines 18-25).

4. Overview of Carley et al.

Carley relates to providing notifications when a plurality of users are altering similar data in a healthcare environment (Title) in order to provide a mechanism to safeguard data (col. 1, line 9). Carley describes that when a load process is initiated to download data from a user station to a server, a check is made for other concurrently executing load processes. If a concurrently executing load process is detected, the users of both processes are notified and one of the processes is suspended. In this way, updates to data can be coordinated so that all desired changes may be made to the data (col. 4, lines 42-64).

Carley contains a discussion of "frameworks" that may be used to design and build a system using notification process (col. 15, lines 48-50). The extensive discussion of these frameworks covers much of columns 16-145.

B. The Rejections Should be Reversed

For reasons described in detail below, the references, even if combined do not teach all limitations of any of the claims and the rejections must be reversed. In summary, none of the references describes updating an component of a collaboration system based on user activity within the collaboration system. Thus, the references do not teach limitations of the claims that relate to the manner in which an update is initiated in a peer-to-peer collaboration system.

1. Independent claim 1 is not properly rejected under 35 U.S.C. §103(a) over Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

The rejection should be reversed because the combination of Donohue and Parthesarathy does not meet all limitations of claim 1. One such limitation not met is the limitation relating to "an activity program."

In the Response to Arguments section of the Final Rejection, the Examiner responded to Applicant's arguments that the references do not show an activity program. The Examiner then cites column 7, lines 12-33 of Donohue as purportedly showing an activity program. As understood based on the cite to column 7, lines 12-33, the Examiner equates the "updater components" of Donohue with the activity program of the present application.

However, the cited passage does not teach an activity program as claimed. Rather, it describes that an updater component can talk to other updater components as described later in that reference (see, column 7, lines 9-10). This passage describes a functionality that allows software identified as "pre-requisite" for software being updated to also be updated (column 13, lines 16-41). Interaction between updater components that update pre-requisite software as described in Donohue does not constitute a peer-to-peer collaboration system that meets the limitations of the claim.

Column 7, lines 12-33 of Donohue does not describe any component that could reasonably be interpreted as an "activity program" as claimed. Specifically, claim 1 recites:

> "an activity program adapted to implement a portion of a collaboration session on a first peer device of the plurality of peer devices, the activity program modifying ***a local data copy of a shared space*** in response to deltas generated as a result of user actions within the collaboration system at the first peer device ***and other peer devices of the plurality of peer devices***, and the activity program generating a component update request in response to an action by a user within the collaboration session, ***the user action indicating a change to the shared space made with the component.***" (emphasis added)

This limitation requires both that the activity program: 1) implement a portion of a collaboration session and 2) generate a component update request in response to an action by a user within the collaboration session indicating a change to the shared space made with the component. The cited passages of Donohue do not describe a component that does either.

An updater component that obtains software updates cannot reasonably be considered to modify "a local data copy of a shared space in response to deltas generated as a result of user actions within the collaboration system." Donohue describes only software updates, not modifying a local data copy of a shared space. As is clear from column 10, lines 1-4, an updater component on one machine only talks to another updater component on another remote machine when the first machine lacks software that is on the other remote machine. In other words, the updaters only interact when there are differences between the machines, not a shared space. Furthermore, there is nothing in Donohue that could reasonably be interpreted as deltas generated as a result of user actions within the collaboration system ... including a change to the shared space." The actions of the updater components in Donohue are clearly described to occur automatically (see, e.g., Abstract), not as a result of user actions.

Moreover, claim 1 recites that the user actions within the collaboration system that give rise to the deltas processed on the first peer device are generated as a result of user action both "at the first peer device and other peer devices of the plurality of peer devices" of the collaboration system. Donohue simply does not describe any updater component that processes deltas generated as a result of user action at two difference devices.

It follows that the updaters of Donohue cannot generate component update requests in response to "user action indicating a change to the shared space made with the component." First, it should be noted that the Office Action cites no portion of Donohue that meets the limitation: "... made with the component." Second, the updates in Donohue occur in response to predefined criteria (col. 15, lines 60-62) and information on software vendor websites (col. 4, lines 62-67). No part of Donohue describes software that is updated in response to a user action indicating a change to a shared space made with the software being updated, which seemingly would be required given the Examiner's interpretation of an updater component as an activity program as claimed.

The Final Rejection also cites to Donohue at column 3, lines 15-21; column 4, lines 15-22 and at column 9, lines 51-54. None of these passages describes a peer-to-peer collaboration system with an activity program as claimed. Instead, these passages merely highlight the errors in equating the updater components of Donohue with an activity program. The cited passage in column 3 describes a prior art software updater utility that checks an online service to find updates. The cited passage at column 4 describes an updater component that performs a comparison between available software updates and installed software on a computer. The cited passage at column 9 indicates that the updater component performs a scan of the operating system to check whether required software resources are already available on a computer.

The Office Action also cites col. 7, line 55 through col. 8, line 12 as teaching an activity program. However, this passage describes "remote server systems" through which software vendors make updates available. The updater components do not maintain the remote server systems. Rather software vendors do (see, e.g., col. 7, lines 61-62; col. 8, lines 31-32). Thus, this passage does not describe anything that reasonably could be regarded as a local data copy of a shared space, and therefore does not describe anything that reasonably could be regarded as an activity program that modifies a shared space in response to user action within a collaboration session, as claimed.

Thus, for multiple reasons, Donohue does not teach an activity program meeting all limitations of the claim. Because the updater components of Donohue do not meet the recited limitations of the claimed activity program, the references, even if combined, would not teach all limitations of claim 1. Parthesarethay is not cited as teaching any of the limitations highlighted

above that are not met by Donohue. Rather, Parthesarethay is only cited as teaching asynchronous retrieval of a file based on a URL. Accordingly, Parthesarethay does not cure the deficiencies of Donohue, and the rejection should be reversed.

2. Dependent claim 2, which depends from claim 1, is not properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Claim 2 depends from claim 1 and is rejected based on the same combination of references, which suffer from the deficiencies described in connection with claim 1, above.

Claim 2 additionally recites that "the system comprises a second activity program installed on the second peer device" and that "the action ***by the user*** [in response to which a component update request is generated] comprises interaction with the second activity program." In rejecting claim 2, the Examiner additionally cites Donohue at col. 3, line 50 – col. 4, line 36. Such a citation is clearly inappropriate for teaching any form of user interaction, let alone the user interaction as recited in claim 2. The cited passage expressly teaches "automatically without requiring any interaction by the user after an initial agreement of update criteria." The passage further states "the user does not need to know where software updates come from, how to obtain them or how to install them since the update component takes care of this." Thus, the rejection is unsupported by any reasonable interpretation of the reference, and the rejection should be reversed.

3. Dependent claim 8, which depends from claim 2, is not properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Claim 8 depends from claim 2 and is rejected based on the same combination of references, which fail to meet the limitations of claim 2 for reasons detailed above.

Claim 8 additionally recites that "the first activity program generates the update request in response to receiving an invitation for a user of the first device to join the collaboration session, the invitation being generated in response to an action by a user of the second device within the

collaboration session." The Office Action cites to Donohue at col. 1, line 51 through col. 2, line 2 and col. 7, lines 22-25 as meeting this limitation. However, no part of these sections of Donohue relates to a user joining a collaboration session or an invitation received by a user of a device. To the contrary, the cited passage in col. 1 and col. 2 relates to software updates distributed over a computer network. The cited passage in col. 7 relates to updater components for distributed software products that can talk to each other.

Thus, the rejection is unsupported by any reasonable interpretation of the reference and the rejection should be reversed.

4. Dependent claim 9, which depends from claim 1, is not properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Claim 9 depends from claim 1 and is rejected based on the same combination of references, which suffers from the deficiencies noted above in connection with claim 1.

Additionally, claim 9 recites that the component [for which a component update request is generated] comprises a tool and the change to the shared space [in response to which a component update request is generated] comprises instantiation of a template for the tool." In rejecting this claim, the Examiner cites to Donohue at col. 7, lines 12-25. No part of the cited passage describes instantiation of a template for a tool in a peer-to-peer collaboration system. To the contrary, the cited passage describes that updater components find and talk to each other through a centralized database.

Thus, the rejection is unsupported by any reasonably interpretation of the reference, and the rejection should be reversed.

5. Independent claim 11 is not properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

In rejecting claim 11, the Office Action cites the same passages of the same references that are cited in connection with claim 1. For reasons detailed above, the Examiner has not properly interpreted Donohue. Consequently, the references, even if combined, would not meet multiple limitations of claim 11.

Claim 11 relates to a computer system that is "part of a peer-to-peer collaboration system comprising at least one other computer system maintaining a first local data copy of a shared space." The claim further recites:

> "generating a component update request ***in response to receiving information about a component used in the collaboration session*** involving the at least one other computer, the component update request identifying the component, and ***the received information being generated dynamically while the collaboration system is in operation to modify the shared space based on input of a plurality of users collaborating using the peer-to-peer collaboration system.***"(emphasis added)

The Office Action cites no passage of Donohue describing generation of a component update request in response to receiving information about a component being used in a collaboration session. As described above in connection with claim 1, Donohue describes generating component update requests based on predetermined criteria and information on software vendor websites and not in response to receiving information about a component being used in a collaboration session. Moreover, the claim requires that the received information in response to which a component update request is generated is itself "generated dynamically while the collaboration system is in operation to modify the shared space based on input of a plurality of users collaborating using the peer-to-peer collaboration system." As noted above in connection with claim 1, Donohue describes automatic updating and does not describe updating in response to user actions, particularly not input of a

plurality of users collaborating using a peer-to-peer collaboration system. Thus, even if Donohue and Parthesarathy were combined, the references would not teach all the limitations of claim 11.

Parthesarathy is not cited to teach the limitations of claim 11 not met by Donohue. Accordingly, even if the references were combined, the combination would not meet the highlighted limitations, and the rejection of claim 11 should be reversed.

6. Dependent claim 12, which depends from claim 11, is not properly rejected under 35 U.S.C. §103(a) based on Donohue et al. (U.S. 6199,204) and Parthesarathy et al. (U.S. 6,353,926).

Claim 12 depends from claim 11, and is rejected based on the same combination of references, which suffer from the deficiencies described above in connection with claim 11.

Additionally, claim 12 recites that "receiving information about a component comprises receiving an invitation to join a collaboration session." In rejecting claim 12, the Examiner cites the same passages of Donohue used in rejecting claim 2. As described above in connection with claim 2, the cited passages of Donohue do not describe an invitation to join a collaboration session. Consequently, there is no mention of generating a component update request in response to receiving an invitation to join a collaboration session, as would be required to meet all limitations of claim 12.

Thus, the rejection is unsupported by any reasonable interpretation of the reference and the rejection of claim 12 should be reversed.

7. Independent claim 21 is not properly rejected under 35 U.S.C. §103(a) based on Varma et al. (U.S. 6,334,141) and Carley et al. (U.S. 6,701,345).

The rejection of independent claim 21 should be reversed because it is based on incorrect interpretation of Varma and Carley. The interpretation of both Varma and Carley posited in the Final Rejection is not supported by the references. When the references are properly interpreted, even if combined, the combination would not meet all limitations of the claim.

As understood, the Final Rejection asserts that Varma at col. 5, line 16 to col. 6, line 27 teaches the following limitation of independent claim 21:

> program code for running a process that maintains a local copy of a shared space, the process being adapted to respond to input from the plurality of members and generates generate a request to update a component used in modifying the local copy of the shared space, the request being generated in response to user input indicating use of the component

However, the cited passage of Varma describes a "workspace modification request" (col. 5, line 36). The passage describes that the workspace modification request is applied to affected partitions of the workspace (col. 5, line 46) while other partitions remain oblivious to the modification (col. 5, line 18). This passage does not describe a component used to generate the modification request. It follows, therefore, that the passage provides no relevant teaching of updating such a component. It follows further that the cited passage provides no relevant teaching of generating a request to update such a component in response to user input indicating use of the component. Consequently, for multiple reasons, Varma does not meet at least the highlighted limitation of claim 21.

Carley does not cure these deficiencies of Varma. In fact, Carley is not cited as teaching any of these limitations. Rather, Carley is cited only as teaching receipt of URL information and asynchronously retrieving files from an identified location. Even if combined, Varma and Carley would not teach all limitations of claim 21, including the limitation highlighted above.

As a further reason that the combination of references does not meet all limitations of the claim, the Final Rejection misinterprets the passages of Carley that are cited as teaching "program code that extracts from the request information which identifies the location of a first file for satisfying the request" and "program code that uses the information to retrieve the first file." In fact, Carley does not teach downloading and installing a component as asserted in the Final Rejection. Col. 50, line 52 – col. 52 line 16 of Carley, cited as teaching this feature, relates to a development tools framework (col. 50, line 18). Carley relates to a data management system (col. 1,

lines 1-10) and specifically is directed to a database system in which multiple users attempt to alter the same data (col. 4, lines 40-45). In context, the cited passage at col. 50, line 52 – col. 52 line 16, is describing tools used to develop such a database system and does not relate to downloading and installing components used during operation of the database system.

Further, some limitations of claim 21 are not addressed in the Final Rejection at all. Neither reference is cited as teaching limitations of claim 21, such as:

> program code that extracts from the first file an indicia of a trusted supplier and obtains second location information of a second file, the second file containing a second component;
>
> program code that uses the second location information to retrieve the second file;
>
> program code to extract from the second file the second component and an indicia of a supplier of the second component; and
>
> program code that selectively installs the second component when the indicia of the supplier is consistent with the indicia of a trusted supplier

The rejection of claim 21 is clearly deficient and should be reversed for any of the foregoing reasons.

8. *Independent claim 32 is not properly rejected under 35 U.S.C. §103(a) based on Varma et al. (U.S. 6,334,141) and Carley et al. (U.S. 6,701,345).*

Independent claim 32 is rejected based on the same incorrect interpretation of Varma and Carley as discussed above in connection with claim 21. Consequently, the cited references do not meet all limitations of claim 32.

For example, neither reference teaches the limitation of claim 32 that recites:

means for implementing a collaboration session for a user, the means for implementing adapted to maintain a local copy of a data space shared by a plurality of collaborating members in the collaboration session and to receive an indication of a component in use within the collaboration session involving the at least one other computer and to selectively generate an update request for the component based on the indication of use within the collaboration session;

Though the Final Rejection cites Varma as meeting this limitation, as described above in connection with claim 21, the cited passage of Varma mentions only that a workspace modification request may exist. There is no mention of a component used to generate the modification request. It follows, therefore that Varma provides no relevant teaching of a request to update such a component, and certainly no such request generated based on an indication of use of the component within a collaboration session. Thus, for multiple reasons, Varma does not teach the highlighted limitation of claim 32.

Moreover, as described above in connection with claim 21, the Final Rejection is based on an improper interpretation of Carley. When properly interpreted, it is clear that Carley does not describe "means for receiving the URL information in asynchronously retrieving the file from the identified location," as claimed.

Because the combination of references, when properly interpreted, does not teach multiple limitations of the claim, the rejection should be reversed.

CONCLUSION

The foregoing identifies errors in each rejection that is appealed. Because of these errors, there is no *prima facie* case of obviousness against any of the claims, and all of the rejections should be reversed.

Dated: July 16, 2009

Respectfully submitted,

By _____
Edmund J. Walsh
Registration No.: 32,950
WOLF, GREENFIELD & SACKS, P.C.
Federal Reserve Plaza
600 Atlantic Avenue
Boston, Massachusetts 02210-2206
617.646.8000

CLAIMS APPENDIX
As Required by 37 CFR 41.37(c)(1)(viii)

Claims Involved in the Appeal of Application Serial No. 10/032,709

1. Apparatus adapted for use in a peer-to-peer collaboration system comprising a plurality of peer devices, the apparatus comprising a computer system with a memory and a computer-readable medium having computer executable modules, the computer-executable modules comprising:

an activity program adapted to implement a portion of a collaboration session on a first peer device of the plurality of peer devices, the activity program modifying a local data copy of a shared space in response to deltas generated as a result of user actions within the collaboration system at the first peer device and other peer devices of the plurality of peer devices, and the activity program generating a component update request in response to an action by a user within the collaboration session, the user action indicating a change to the shared space made with the component;

a component manager that receives the component update request from the activity program and has a parser that extracts from the request URL information which identifies the location of a file containing software component resources for satisfying the component update request;

a download manager that receives the URL information from the component manager and has a file retriever which asynchronously retrieves the file from the specified location and places the file in a staging area in the memory; and

an install manager that asynchronously installs the file,

wherein the component manager is adapted to determine whether the requested software component is already installed on the computer system and to selectively invoke the download manager based on the determination.

2. The apparatus of claim 1, wherein:

the peer-to-peer collaboration system further comprises a second peer device; and

the activity program comprises a first activity program and is installed on the first peer device;

the system comprises a second activity program installed on the second peer device; and

the action by the user comprises interaction with the second activity program.

3. The apparatus of claim 1 wherein the component manager comprises a security section that validates the file before installation.

4. The apparatus of claim 1 further comprising a manifest which contains a list of all software components installed on the computer system.

5. The apparatus of claim 4 wherein the component manager comprises a mechanism that responds to the request by checking the manifest to ascertain whether the requested software component is already installed on the computer system.

6. The apparatus of claim 1 wherein the component manager comprises a polling mechanism that periodically polls component locations to locate new component versions.

7. The apparatus of claim 5 wherein the software component may be a system component that is required for operation of the apparatus or an application component that is not required for operation of the apparatus and wherein the apparatus further comprises a system component manager that receives a request for the system component and a system component installer that is started by the system component manager.

8. The apparatus of claim 2 wherein the first activity program generates the update request in response to receiving an invitation for a user of the first device to join the collaboration session, the invitation being generated in response to an action by a user of the second device within the collaboration session.

9. The apparatus of claim 1 wherein the component comprises a tool and the change to the shared space comprises instantiation of a template for the tool.

10. The apparatus of claim 7 wherein the component manager comprises an activation factory for activating installed software components.

11. A method of operating a computer system with a memory as part of a peer-to-peer collaboration system comprising at least one other computer system maintaining a first local data copy of a shared space, the method comprising:

(a) generating a component update request in response to receiving information about a component being used in a collaboration session involving the at least one other computer, the component update request identifying the component, and the received information being generated dynamically while the collaboration system is in operation to modify the shared space based on input of a plurality of users collaborating using the peer-to-peer collaboration system;

(b) parsing the request to extract from the request URL information which identifies the location of a file containing software component resources for satisfying the request and an identification of the component;

(c) determining, based on the identification of the component, availability of the component;

(d) selectively in response to the determination, using the URL information to asynchronously retrieve the file from the specified location; and

(e) asynchronously installing the component from the file, whereby the computer system can use the component to maintain a second local data copy of the shared space that is synchronized with the first local data copy.

12. The method of claim 11 wherein receiving information about a component comprises receiving an invitation to join a collaboration session.

13. The method of claim 11 further comprising:

(f) validating the file before installation.

14. The method of claim 11 further comprising:

(f) operating the computer system to implement a portion of the collaboration session using the component.

15. The method of claim 11 wherein step (c) comprises checking a manifest to ascertain whether the requested software component is already installed on the computer system before retrieving the file.

16. The method of claim 11 further comprising:

(f) periodically polling component locations to locate new component versions.

17. The method of claim 11 wherein the component may be a system component that is required for operation of the apparatus or an application component that is not required for operation of the apparatus and wherein the method further comprises:

(f) when the component is a system component, installing the component with a separate system component manager, which receives a request for a system component, and a separate system component installer that is started by the system component manager.

18. The method of claim 17 wherein step (f) comprises shutting the system component manager down before installing in-use components.

19. The method of claim 18 wherein step (f) further comprises restarting the system component manager after the system component has been installed.

20. The method of claim 11 further comprising:

(f) activating the installed software component with an activation factory.

21. A computer program product stored on a computer readable medium for use in a peer-to-peer collaboration system including a computer system with a memory, the computer system being adapted to receive an update request generated as a result of user interaction with the peer-to-peer collaboration system by a plurality of members of a collaboration session, the computer program product comprising:

program code for running a process that maintains a local copy of a shared space, the process being adapted to respond to input from the plurality of members and generate a request to update a component used in modifying the local copy of the shared space, the request being generated in response to user input indicating use of the component;

program code that extracts from the request information which identifies the location of a first file for satisfying the request;

program code that uses the information to retrieve the first file;

program code that extracts from the first file an indicia of a trusted supplier and obtains second location information of a second file, the second file containing a second component;

program code that uses the second location information to retrieve the second file;

program code to extract from the second file the second component and an indicia of a supplier of the second component; and

program code that selectively installs the second component when the indicia of the supplier is consistent with the indicia of a trusted supplier.

22. The computer program product of claim 21 wherein the first file contains an OSD description of the software component, including dependent components.

23. The computer program product of claim 21 wherein the program code that retrieves the second file is the same program code that retrieves the first file.

24. The computer program product of claim 21 wherein the program code that extracts from the first file the indicia of a trusted supplier extracts a fingerprint of a trusted supplier.

25. The computer program product of claim 21 wherein the program code that uses the information to retrieve the file from the specified location comprises program code that checks a manifest to ascertain whether the requested software component is already installed on the computer system before retrieving the file.

26. The computer program product of claim 21 wherein the program code that selectively installs the second component selectively prompts a user for authorization to install the second component when the indicia of the supplier is not consistent with the indicia of a trusted supplier.

27. The computer program product of claim 21 wherein the software component may be a system component that is required for operation of the apparatus or an application component that is not required for operation of the apparatus and wherein the computer program product further comprises program code that installs a system component with a separate system component manager that receives a request for the system component and a separate system component installer that is started by the system component manager.

28. The computer program product of claim 27 wherein the program code that installs system components with a separate system component manager and a separate system component installer comprises program code that shuts the system component manager down before installing in-use components.

29. The computer program product of claim 28 wherein the program code that installs the system component with a separate system component manager and a separate system component installer further comprises program code that restarts the system component manager after the system component has been installed.

30. The computer program product of claim 21 further comprising program code that activates installed software components with an activation factory.

32. Apparatus for use in a peer-to-peer collaboration system comprising a computer system with a memory and at least one other computer system maintaining a first local data copy of a shared space, the apparatus comprising:

means for implementing a collaboration session for a user, the means for implementing adapted to maintain a local copy of a data space shared by a plurality of collaborating members in the collaboration session and to receive an indication of a component in use within the collaboration session involving the at least one other computer and to selectively generate an update request for the component based on the indication of use within the collaboration session;

means responsive to the request, for parsing the request to extract from the request URL information which identifies the location of a file containing software component resources for satisfying the request;

means for receiving the URL information and asynchronously retrieving the file from the identified location; and

means cooperating with the parsing means for installing the component from the file while the collaboration system is executing, whereby the computer system can use the component to maintain a second local data copy of the shared space that is synchronized with the first local data copy.

33. The apparatus of claim 32 further comprising means for selectively deferring installation of the component until the collaboration system is not executing.

34. The apparatus of claim 32 wherein the parsing means comprises means for validating the file before installation.

35. The apparatus of claim 32 further comprising a manifest which contains a list of all software components installed on the computer system.

36. The apparatus of claim 35 wherein the parsing means comprises means responsive to the request for checking the manifest to ascertain whether the requested software component is already installed on the computer system.

37. The apparatus of claim 32 wherein the parsing means comprises means for periodically polling component locations to locate new component versions.

38. The apparatus of claim 32 wherein the software component may be a system component that is required for operation of the apparatus or an application component that is not required for operation of the apparatus and wherein the apparatus further comprises a system component manager that receives a request for the system component and a system component installer that is started by the system component manager.

39. The apparatus of claim 38 wherein the system component installer comprises means for shutting the system component manager down before installing an in-use component.

40. The apparatus of claim 39 wherein the system controller comprises means for restarting the system component manager after system component has been installed.

41. The apparatus of claim 32 wherein the parsing means comprises means for activating installed software components.

EVIDENCE APPENDIX

As Required by 37 CFR 41.37(c)(1)(ix)

No evidence pursuant to §§ 1.130, 1.131, or 1.132 or entered by or relied upon by the Examiner is being submitted.

RELATED PROCEEDINGS APPENDIX
As Required by 37 CFR 41.37(c)(1)(x)

No related proceedings are referenced in II. above, hence copies of decisions in related proceedings are not provided.